EE 6882
Visual Search Engine
Feb. 27th, 2012
Lecture #6
- Object Search Using Local Features
- Applications of Mobile Visual Search
- Mid-Term Project

Reading List
Local Appearance Descriptor (SIFT)

[Lowe, ICCV 1999]

Histogram of oriented gradients over local grids
- e.g., 4x4 grids and 8 directions
  -> 4x4x8=128 dimensions
- Rotation-aligned, Scale invariant

There are many other local features, e.g., SUFR, HOG, BRIEF, MSER, STIP

Example of Local Feature Matching

Slide credit: J. Sivic
Application: Large Scale Mobile Visual Search

Mobile Visual Search

1. Take a picture
2. Send image or features
3. Send to server via MMS
4. Feature matching with database images
5. Send results back
Application: particular object retrieval

Example I: Visual search in feature films

Visually defined query

“Find this clock”

“Groundhog Day” [Rammis, 1993]

Example II: Search photos on the web for particular places

Find these landmarks ...in these images and 1M more

Slide credit: J. Sivic
Global vs. Local Feature Matching

- **Global**
  - Convert query and database images to global representations such as Bags of Words
  - Perform global matching

- **Local**
  - Use each local feature as query
  - Search matched local features in the database
  - Rank images
  - Perform spatial verification

Outline of a local feature retrieval strategy

1. Compute affine covariant regions in each frame independently
2. "Represent" each region by a vector of descriptors
3. Finding corresponding regions is transformed to finding nearest neighbour vectors
4. Rank retrieved frames by number of corresponding regions
5. Verify retrieved frame based on spatial consistency

Slide credit: J. Sivic
Bottleneck: nearest-neighbor matching over gigantic database

Solve following problem for all feature vectors, \( x_j \), in the query image:

\[
\forall j \quad NN(j) = \arg \min_i ||x_i - x_j||
\]

where \( x_i \) are features in database images.

Nearest-neighbour matching is the major computational bottleneck

- Linear search performs \( dxn \) operations for \( n \) features in the database and \( d \) dimensions
- \( n \) may be as high as billions
- No exact methods are faster than linear search for \( d>10 \)
- Explore approximate methods (e.g., tree based indexing)

K-d tree construction

Simple 2D example
K-d tree query

Approximate nearest neighbour K-d tree search

**Issues**

- Need backtracing to find exact NN
- Exponential cost when dimension grows
- Remedy: limit the number of neighboring bins to explore
- Search k-d tree bins in order of distance from query
Alternative method: mapping local features to **Visual Words**

Visual words: main idea

*Extract some local features from a number of images …* 

e.g., SIFT descriptor space: each point is 128-dimensional

Slide credit: D. Nister
Visual words: main idea

Slide credit: D. Nister
Visual words: main idea

Slide credit: D. Nister
Visual words: main idea

Map high-dimensional descriptors to tokens/words by quantizing the feature space

- Quantize via clustering, let cluster centers be the prototype “words”
Visual Words: Image Patch Patterns

Corners  
Blobs  

eyes  
letters

Inverted file index for images comprised of visual words

- Score each image by the number of common visual words (tentative correspondences)
- But: does not take into account spatial layout of regions
How to create visual words?
Clustering / quantization methods

- k-means (typical choice), agglomerative clustering, mean-shift,…

- Hierarchical clustering: allows faster insertion / word assignment while still allowing large vocabularies
  - Vocabulary tree [Nister & Stewenius, CVPR 2006]

Quantization using K-means

- K-means overview:

  - K-means provably locally minimizes the sum of squared errors (SSE) between a cluster centre and its points

  - But: The quantizer depends on the initialization.

  - The nearest neighbour search is the bottleneck
Approximate K-means

- Use the approximate nearest neighbour search (randomized forest of kd-trees) to determine the closest cluster centre for each data point.

- Original K-means complexity = $O(N K)$
- Approximate K-means complexity = $O(N \log K)$
- Can be scaled to very large $K$.

Clustering / quantization methods

- k-means (typical choice), agglomerative clustering, mean-shift, …

- Hierarchical clustering: allows faster insertion / word assignment while still allowing large vocabularies
  - Vocabulary tree [Nister & Stewenius, CVPR 2006]
Example: Recognition with Vocabulary Tree

Tree construction:

Vocabulary Tree

Training: Filling the tree
Vocabulary Tree
Training: Filling the tree

[Nister & Stewenius, CVPR’06]
Slide credit: David Nister
Vocabulary Tree

Recognition

Verification on spatial layout

Voc Tree can also be used to score images efficiently

q: query feature vector
d: database feature vector

$$\| q - d \|_p^2 = \sum_i |q_i - d_i|^p$$

$$= \sum_{i|q_i = 0} |q_i|^p + \sum_{i|q_i \neq 0, d_i \neq 0} |q_i|^p + \sum_{i|q_i \neq 0, d_i = 0} |q_i - d_i|^p$$

$$= \| q \|_p^p + \| d \|_p^p + \sum_{i|q_i \neq 0, d_i \neq 0} (|q_i - d_i|^p - |q_i|^p - |d_i|^p)$$

$$= 2 + \sum_{i|q_i \neq 0, d_i \neq 0} (|q_i - d_i|^p - |q_i|^p - |d_i|^p).$$

Incremental update of scores for every query visual word
Vocabulary Tree: Performance

Evaluated on large databases
- Indexing with up to 1M images

Online recognition for database of 50,000 CD covers
- Retrieval in ~1s

Find experimentally that large vocabularies can be beneficial for recognition

[Nister & Stewenius, CVPR’06]

Beyond Bag of Words

- Use the position and shape of the underlying features to improve retrieval quality

- Both images have many matches – which is correct?
Beyond Bag of Words

- We can measure **spatial consistency** between the query and each result to improve retrieval quality

Many spatially consistent matches – **correct result**

Few spatially consistent matches – **incorrect result**

---

Beyond Bag of Words

- Extra bonus – gives **localization** of the object
Spatial Verification

- Check consistency of relative distance (shift)
- Check consistency of scale change
- Check consistency of transformation (RANSAC)

Feature-space outlier rejection

Can we now compute H from the blue points?
- No! Still too many outliers…
- What can we do?
RANSAC for estimating homography

RANSAC loop:
1. Select four feature pairs (at random)
2. Compute homography $H$ (exact)
3. Compute inliers where $SSD(p_i', H p_i) < \epsilon$
4. Keep largest set of inliers
5. Re-compute least-squares $H$ estimate on all of the inliers
Estimating spatial correspondences

1. Test each correspondence

Estimating spatial correspondences

2. Compute a planar affine transformation (6 dof)

Need just one correspondence
Estimating spatial correspondences

3. Score by number of consistent matches

Re-estimate full affine transformation (6 dof)

Verification by spatial layout - overview

1. Query

2. Initial retrieval set (bag of words model)

3. Spatial verification (re-rank on # of inliers)

Slide credit: J. Sivic
Oxford buildings dataset

- Automatically crawled from **flickr**
- Consists of:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Resolution</th>
<th># images</th>
<th># features</th>
<th>Descriptor size</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>1024 × 768</td>
<td>5,062</td>
<td>16,334,970</td>
<td>1.9 GB</td>
</tr>
<tr>
<td>ii</td>
<td>1024 × 768</td>
<td>99,782</td>
<td>277,770,833</td>
<td>33.1 GB</td>
</tr>
<tr>
<td>iii</td>
<td>500 × 333</td>
<td>1,040,801</td>
<td>1,186,469,709</td>
<td>141.4 GB</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td>1,145,645</td>
<td>1,480,575,512</td>
<td>176.4 GB</td>
</tr>
</tbody>
</table>

- Landmarks plus queries used for evaluation

- Ground truth obtained for 11 landmarks
- Evaluate performance by mean Average Precision
Query Expansion in text

In text:
- Reissue top n responses as queries
- Pseudo/blind relevance feedback
- Danger of topic drift

In vision:
- Reissue spatially verified image regions as queries

Query expansion in text - example

Original query: Hubble Telescope Achievements

Query expansion: Select top 20 terms from top 20 documents

Example from: Jimmy Lin, University of Maryland
Automatic query expansion

Visual word representations of two images of the same object may differ (due to e.g. detection/quantization noise) resulting in missed returns

Initial returns may be used to add new relevant visual words to the query

Strong spatial model prevents ‘drift’ by discarding false positives

[Chum, Philbin, Sivic, Isard, Zisserman, ICCV’07]
Query Expansion

Query Image

Originally retrieved image

Originally not retrieved

Slide credit: J. Sivic

Query Expansion

Slide credit: J. Sivic
Query Expansion

New expanded query is formed as:
- the average of visual word vectors of spatially verified returns
- only inliers are considered
- regions are back-projected to the original query image
Issues of Tree-based Indexing

- Robust visual search requires large trees
  - > 1 million nodes
  - Tree index is compact – 20 bits per feature
  - But difficult to store large trees at mobile clients
- Another problem:
  - In high-dimensional space, back-tracing for NN search becomes a bottleneck
- Codeword index does not preserve proximity

**Slide credit:** J. Sivic
Alternative: Hashing as Compact Code

- Locality Sensitive Hashing (LSH) [Indyk & Motwani 98]

\[ x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \]
\[ h_1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \]
\[ h_2 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \]
\[ \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \]
\[ h_k \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \]

- Solve approximate NN problem with probabilistic guarantee

For any two points p, q:
- if \(|p-q| < r\) then \(Pr[h(p) = h(q)] > P_1\)
- if \(|p-q| > cr\) then \(Pr[h(p) = h(q)] < P_2\)

Figure: J. Sivic

Hash based image retrieval
Beyond LSH: Compact and Balanced

- Pure random projections lead to redundancy and imbalance
- SPICA Hash (He et al CVPR 11): jointly optimizes search accuracy & time

\[ D(Y) = \sum_{p,q=1}^{N} W_{pq} \| Y_p - Y_q \|^2 \leq \eta \]

Locality sensitive

\[ \min I(y_1, \ldots, y_M) \]

while \[ E(y) = \sum_{p=1}^{N} Y_p = 0 \]

Balanced bucket size

Beyond LSH: Compact and Balanced

- SPICA Hash: jointly optimize search accuracy & time
- Results

The largest LSH bucket contains 10% of points

Reduce candidate set from 1M to 10K @ 0.5 recall
Beyond Randomness: Semi-Supervised Learned Projection

Wang, Kumar, and Chang, CVPR, ICML 2010

- Given Pair-Wise Relations
  \[(x_i, x_j) \in \mathcal{M}: \text{neighbor pair}
  (x_i, x_j) \in \mathcal{C}: \text{nonneighbor-pair}\]

- Measure empirical fit of hash bits

- Are the partitions balanced?
  Measure hash bit variance

- Elegant eigen-decomposition solution
  \[
  J(W) = \frac{1}{2} \text{tr} [W^T X S X^T W] + \frac{\eta}{2} \sum_k E[||w_k^T x||^2]
  \]

- Incremental learning via AdaBoosting

- Learned projection hash increases accuracy > 2X
- Query time: a few seconds
- Compact code - 48 bits vs. 128 bytes per sample)
- Challenge: scale to billions

Tiny Image – 80M
Columbia Mobile Product Search using Bags of Hash Bits (BoHB)

Additional Feature: Boundary
- Use automatic salient object segmentation for every image in DB [Cheng et al, CVPR 2011]
- Boundary features:
  - normalized central distance, Fourier magnitude
- Invariance: translation, scaling, rotation
Boundary Feature – Central Distance

- Sample N points \( p(n) \) with equal distance along boundary
- Compute the distance from every sample point \( p(n) \) to the boundary center \( c \)
- Normalized the distance vector by the maximal distance

\[
D(n) = \frac{\|p(n) - c\|_2}{\max_n \|p(n) - c\|_2}
\]

- Apply FFT on the distance vector, and take the magnitude part

\[
F(n) = |f[D(n)]|
\]

- Invariance: translation, scale, rotation
Reranking with boundary feature

Columbia MPS System:
Bags of Hash Bits and Boundary features

**Server:**
- 400,000 product images crawled from Amazon, eBay and Zappos
- Hundreds of categories: shoes, clothes, electrical devices, groceries, kitchen supplies, movies, etc.

**Speed**
- Feature extraction: ~1s
- Transmission: 80 bits/feature, 1KB/im
- Server Search: ~0.4s
- Download/display: 1-2s

[video demo link]
Multi-View Challenge

How to guide the user to take a successful mobile query?
- Which view will be the best query?
  • For example, in mobile location search:
  • Or in mobile product search:

Solution: Active Query Sensing

- Guide User to a More Successful Search Angle
- Active Query Sensing [Yu, Ji, Zhang, and Chang, ACM MM '01]
Active Query Sensing System

Basic Idea and Workflow

• Offline
  – Salient view learning for each reference location
• Online
  – Viewing angle prediction of the first query
  – Suggest new views by majority voting

Predict the view angle of query

Offline Training: Train view prediction classifiers offline

Online Prediction: View alignment based on the image matching

Our solution is to combine them both
Active Query Sensing

- Find the most reliable view angle and suggest angle change

Salient View (Offline)

Viewing Angle Prediction

View Change

Turn 90 degrees to the right

User Interface

- Help user determine whether the first query is correct
  - Panorama
  - Geographical map context
- Guide the user to take the second query
  - Compass, camera icon
- Show point of interest
Mid-Term Project

Possible Topics (not limited to these):

- Review and test new features (local features, sketch, depth, etc.)
- Review and test multi-feature fusion methods
- Visual search of specific types of data
  - Patent diagrams, fashion, consumer, street view, nature, etc.
- Visual summaries: panorama, photo poster, slide show
- New user interfaces (search by sketch, gesture, voice)

Mobile Search Project

- Tools and systems on mobile devices
- Real-time feature extraction and tracking from images/videos, PTAM1, PTAM2
- Salient object detection and indexing (remember where the objects are UCLA project)
- Real-time object detection, text detection, OCR
- Combine GPS and local information
Mid-Term Project

- Focus on reviews in mid-term projects (testing welcome)
  - expand to the final project with new ideas and experiments
- OK to use data in HW#1 and #2
- Two persons in a team, but exceptions considered.
- 3/5: 1-page proposal due (summary, work plan, & references)
- 3/26: mid-term report and narrated slides (15 mins)
- 4/30: final project report due 4/30